

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁶ : G06F 7/72	A1	(11) International Publication Number: WO 99/04332
		(43) International Publication Date: 28 January 1999 (28.01.99)

(21) International Application Number: PCT/IL98/00327

(22) International Filing Date: 13 July 1998 (13.07.98)

(30) Priority Data:
121297 14 July 1997 (14.07.97) IL

(71) Applicant (for all designated States except US): L.P.K. – INFORMATION INTEGRITY LTD. [IL/IL]; Sigalon Street 38, 84965 Omer (IL).

(72) Inventor; and
(75) Inventor/Applicant (for US only): ARAZI, Benjamin [IL/IL]; Sigalon Street 38, 84965 Omer (IL).

(74) Agents: LUZZATTO, Kfir et al.; Luzzatto & Luzzatto, P.O. Box 5352, 84152 Beer-Sheva (IL).

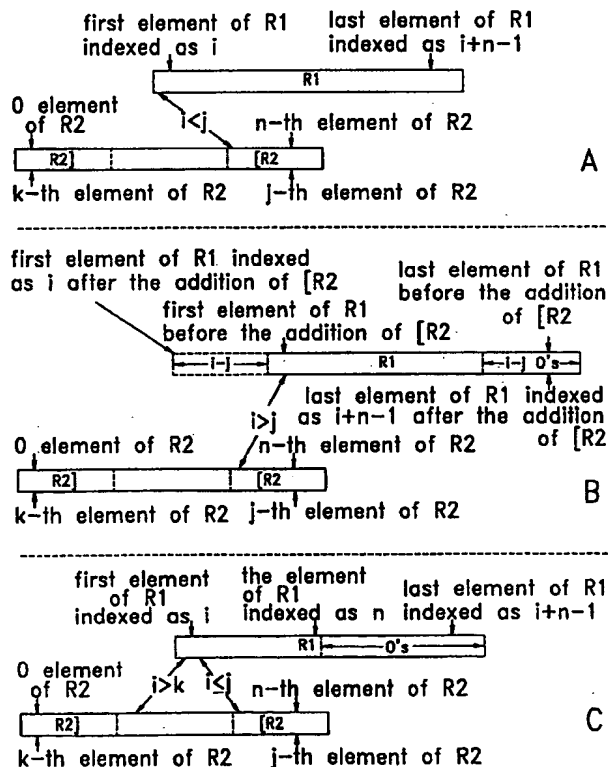
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: COMPOSITE FIELD MULTIPLICATIVE INVERSE CALCULATION FOR ELLIPTIC CURVE CRYPTOGRAPHY

(57) Abstract

A method for calculating modular multiplicative inverses of a Galois Field $GF((2^m)^n)$ comprising the steps of: having first (R0), second (R2) and third (R1) memory units, storing respectively, $n+1$, $n+1$ and n elements of the field $GF(2^m)$; storing in said first memory unit the generating polynomial ($g(x)$) of the external field of said Galois Field; storing in said second memory unit the field element to be inverted, followed with a 0; performing various adding, shifting and multiplication operations on these registers in the lines of the Euclid algorithm; whereby: a) to reduce the contents of said first part of said second memory unit in the lines of the Euclid algorithm by canceling elements both from the most significant location and from the least significant location of said first part of said second memory unit; and b) to calculate modular multiplicative inverses by effecting steps in which a multiplication of a plurality of elements of the field $GF(2^m)$ by one element of said field is always effected when said plurality of elements is stored in one specific memory unit (R2) of a fixed length, without a prior exchange of contents between said one specific memory unit and another memory unit.



BEST AVAILABLE COPY

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakhstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

BEST AVAILABLE COPY

COMPOSITE FIELD MULTIPLICATIVE INVERSE CALCULATION FOR ELLIPTIC CURVE CRYPTOGRAPHY

Field of the Invention

The present invention relates to a method and apparatus for efficiently implementing elliptic curve cryptographic operations over the Galois field $GF((2^m)^n)$ based on an efficient software or hardware operator which calculates modular multiplicative inverses and the product and the square of elements of said field.

Background of the Invention

Elliptic Curve Cryptography (ECC) is one of the modern approaches for the implementation of key exchange over open channels and the generation of digital signatures. The underlying principles of ECC were published in N. Koblitz, "Elliptic Curve Cryptosystems", Mathematics of Computation, 48, pp. 203-209, 1987 and in V. Miller, "Uses of Elliptic Curves in Cryptography", Crypto '85, Springer-Verlag LNCS 218, pp. 417-426, 1986. ECC techniques and implementations are specified in ANSI X9.62-199x, Working Draft, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm, January 15, 1997, and in IEEE P1363 Working Draft, February 6, 1997.

Three fundamental arithmetic operations are required for implementing ECC: a) the calculation of modular multiplicative inverses of finite field elements; b) modular multiplication of such elements and c) their modular squaring. This invention concerns fast and efficient software and hardware methods for executing these operations in the arithmetic field known as the Galois field $GF((2^m)^n)$. ECC implementations over Galois field were indicated in G. Harper, A. Menezes and V. Vanstone, "Public Key Cryptosystems with Very Small Key Lengths", Eurocrypt '92, LNCS 658 pp. 163-173 and in De Win et al, "A Fast Software Implementation for Arithmetic Operations in $GF(2^n)$ ", Asiacrypt '96.

The advantages in implementing ECC-related operations over said $GF((2^m)^n)$ stem from the ability to process m -bit values at a time, rather than single-bit values. Choosing values for m like 8, 16 or 32 are especially suitable for current CPUs.

Two polynomials with binary coefficients are used when operating over said $GF((2^m)^n)$. One polynomial, $g(x)$, of degree n , defines an 'external field'. Another polynomial, $f(x)$, of degree m , defines an 'internal field'. n and m should be relatively prime. The elements of said $GF((2^m)^n)$ are polynomials of degree $n-1$ or less, which involve operations over said polynomial $g(x)$. The coefficients of these polynomials are themselves binary polynomials of degree $m-1$ or less, which involve operations in the field $GF(2^m)$ over said polynomial $f(x)$. Operations over said $GF((2^m)^n)$ can be executed either over the polynomial basis or the normal basis, as is clear to persons skilled in the art.

Hereinafter, R_{ij} denotes an element of the field $GF(2^m)$ stored in the j -th place in memory unit R_i , wherein the index of the first (left) element is 0. The index i can have the values 0, 1 or 2. That is, the notation R_{ij} refers to elements stored in memory units R_0 , R_1 or R_2 .

Furthermore, forthcoming descriptions of operations over the field $GF((2^m)^n)$ suit the case in which the polynomial $g(x)$ is a trinomial in which s is the index of the middle non-zero coefficient. Extensions to a general polynomial $g(x)$ will be clear to persons skilled in the art.

A method for effecting the operations of multiplying two elements of the field $GF((2^m)^n)$ is described in the following Pseudo-code 1. A further explanation follows the description.

Pseudo-code 1: Multiplying two elements of the field $GF((2^m)^n)$

R0 and R2 are memory units that store $n+1$ elements of the field $GF(2^m)$, while memory units R1 stores n elements.

Initially: R0 contains the multiplier $b(x)$ of the field $GF((2^m)^n)$ followed by a 0;

R2 contains the multiplicand $c(x)$ of said field followed by a 0;

R1 contains 0's.

(comment: Said right-most 0 in R0 and R2 is the 0 element of the field $GF(2^m)$.)

for $i = 0$ to $n-2$

$R1 = R1 + R2 * R0i$

(comment: The operation $R2 * R0i$ and the subsequent addition to R1 does not concern the right-most element in R2.)

shift-right R2

$R20 = R2n$

$R2s = R2s + R2n$

(comment: The above right-shift multiplies the contents of R2 by x , modulo the generating trinomial $g(x)$.)

$R1 = R1 + R2 * R0(n-1)$

stop

The final content of R1 is $b(x) \cdot c(x)$.

Explanation:

The above process is a shift-and-add process, in which the right shift operations of R2 modulo the generating polynomial generate successive values of $x^i \cdot c(x)$. These values are multiplied by the corresponding coefficient of $b(x)$ and the results of said multiplications are accumulated in R1.

A method for effecting the operation of squaring an element of the field $GF((2^m)^n)$ is described in the following Pseudo-code 2. A further explanation follows the description.

Pseudo-code 2: Squaring an element of the field $GF((2^m)^n)$

R1 is a memory unit that stores n elements of the field $GF(2^m)$, while memory unit R2 stores $n+1$ elements.

Initially: R1 contains the element $b(x)$ to be squared;
R2 contains 0's.

for $i = 0$ to $n-1$

$$R1i = [R1i]^2$$

for $i = 1$ to $(n-1)/2$

$$R2(n-1) = R2(n-1) + R1(n-1)$$

shift-right R1 from index $(2i-1)$

(*Comment:* after this stage, the element of the field $GF(2^m)$ that was originally at location i in memory unit R1 is now at location $2i$ and the element that was originally at location $n-1$ is shifted out of the memory unit.)

2x	shift-right R2
	$R20 = R2n$
	$R2s = R2s + R2n$

(*Comment:* The execution of the above three lines is repeated twice. The first execution processes the element moved from R1 as $x^n \bmod g(x)$. The second execution ensures that finally this coefficient is placed, mod $g(x)$ at a place with a doubled index. When $i = (n-1)/2$ the elements stored in R1 stand for the coefficients of $b(x)^2$ of degrees $n-1$ or less. The following operation adds this lower part of $b(x)^2$ to R2.)

$R2 = R2 + R1$

stop

The final content of R2, while ignoring the right-most (n-th) element, is $b(x)^2$

Explanation:

This method is based on the fundamental fact that the square of $b(x)$ is obtained by taking the square over the field $GF(2^m)$ of each of the coefficients of $b(x)$ and then placing each of the coefficients at a location whose index is the double of the original index, where placements in indices higher than $n-1$ are accompanied with operations modulo the generating polynomial $g(x)$ of the field $GF((2^m)^n)$.

When operating over the normal basis of the field of $GF(2^m)$, said operation $R1i = [R1i]^2$ indicated in Pseudo Code 2, is effected as a single cyclic shift of the field element, as will be clear to persons skilled in the art. It would be possible to simultaneously cyclically shift all n elements of the field $GF(2^m)$ stored within said memory unit R1.

Modular multiplicative inverse operations, needed to be executed when implementing ECC operations, can be based on exponentiations such as indicated in G.B. Agnew et al., "An Implementation of Elliptic Curve Cryptosystems over F_2^{155} ", IEEE J. on Sel. Areas in Communications, 1993, pp. 804-813, or on the Euclid algorithm. Euclid-based calculations of the multiplicative inverse of an element of the field $GF(2^n)$ are shown in E.R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, 1968, pp. 36-44. Euclid-

based calculations of the multiplicative inverse of an element of $GF((2^m)^n)$ are shown in De Win et al., "A Fast Software Implementation for Arithmetic Operations in $GF(2^n)$ ", Asiacrypt '96 and in R. Schroeppe et al., "Fast Key Exchange with Elliptic Curve Systems", Crypto '95, LNCS 963, 1995, pp. 43-56.

Summary of the Invention

The invention relates to a first method for calculating modular multiplicative inverses of a Galois Field $GF((2^m)^n)$, by an apparatus having first, second and third memory units, the method comprising the steps of:

- having first (R0), second (R2) and third (R1) memory units storing respectively, $n+1$, $n+1$ and n elements of the field $GF(2^m)$;
- storing in said first memory unit the generating polynomial ($g(x)$) of the external field of said Galois Field;
- storing in said second memory unit the field element to be inverted, followed with a 0;
- adding the contents of the first part of said second memory unit (R2]) to the contents of said first memory unit and adding the contents of the second part of said second memory unit ([R2) to the contents of said third memory unit in the lines of the Euclid algorithm;
- adding the contents of a part of said first memory unit to the contents of the first part of said second memory unit (R2]) and adding the contents of a part of said third memory unit to the contents of the second part of said second memory unit ([R2) in the lines of the Euclid algorithm and in order to convert into a zero value the element of the field $GF(2^m)$ stored at the most significant place at said first part of said second memory unit;
- effecting shift operations on said first memory unit and said first part of said second memory unit, while not shifting said second part of said second memory unit, in order to cancel elements of the field $GF(2^m)$ stored at the least significant place at said first and second memory units;
- effecting multiplication operations in which the entire contents of said second memory unit are multiplied by one element of said field in the lines of the Euclid algorithm;

whereby:

a. to reduce the contents of said first part of the second memory unit in the lines of the Euclid algorithm by canceling elements both from the most significant location and from the least significant location of said first part of second memory unit; and

b. to calculate modular multiplicative inverses by effecting steps in which a multiplication of a plurality of elements of the field $GF(2^m)$ by one element of said field is always effected when said plurality of elements is stored in one specific memory unit (R2) of a fixed length, without a prior exchange of contents between said one specific memory unit and another memory unit.

According to one embodiment of the invention, each one of said elements of the field $GF(2^m)$ is a single bit.

The invention further relates to a second method for effecting the operations of calculating the product of two elements of the field $GF((2^m)^n)$, calculating the square of an element of said field and calculating the multiplicative inverse of an element of said field, comprising the steps of:

- having first (R0), second (R2) and third (R1) memory units which store, respectively, $n+1$, $n+1$ and n elements of the field $GF(2^m)$;
- multiplying all the elements of the field $GF(2^m)$ stored in said second memory unit by one element of said field where no such multiplication is effected in connection with said first or third memory units and where no exchange of contents between said second memory unit and said first or third memory units takes place prior to said multiplication;
- adding a variable number of elements of the field $GF(2^m)$ stored in said first memory unit to elements stored in said second memory unit;
- adding a variable number of elements of the field $GF(2^m)$ stored in said second memory unit to elements stored in said first memory unit;

- adding a variable number of elements of the field $GF(2^m)$ stored in said third memory unit to elements stored in said second memory unit;
- adding a variable number of elements of the field $GF(2^m)$ stored in said second memory unit to elements stored in said third memory unit;
- shifting, either physically or by manipulations with indices, the elements stored in said first memory unit;
- shifting, either physically or by manipulations with indices, a variable number of elements stored in said second memory unit;
- shifting, either physically or by manipulations with indices, the elements stored in said second memory unit while modular operations over the generating polynomial of the field $GF((2^m)^n)$ are being performed; and
- shifting, either physically or by manipulations with indices, a variable number of elements stored in said third memory unit;

whereby to have a unified method which effects, by effecting any of the aforesaid steps, the operations of calculating the product of two elements of the field $GF((2^m)^n)$, calculating the square of an element of said field and calculating the multiplicative inverse of an element of said field.

Preferably, the value of said m equals the product of two different prime numbers p and r and the calculation of the modular multiplicative inverses is performed by processing p -bit values at a time.

The invention further relates to an apparatus for carrying out said first method, in which a multiplication of a plurality of elements of the field $GF(2^m)$ which is stored in said specific memory unit, by one element of said field which is stored in a buffer memory, is effected by using a plurality of circuits, each such circuit multiplying one element by an element stored in said specific memory unit. Preferably, in said apparatus the multiplication of a plurality of elements of the field $GF(2^m)$ stored in said specific memory unit by one element of said field is effected by using one pre-processing circuit which processes said one element and a plurality of post-processing circuits, each such circuit operating on two values, one being the output of

said pre-processing circuit and the other value being an element stored in said specific memory unit.

Furthermore, the invention relates to an apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, which comprises: a buffer memory for storing one element of the field $GF(2^m)$, a first memory unit (R0) for storing $n+1$ elements of the field $GF(2^m)$, a second memory unit (R2) for storing $n+1$ elements of the field $GF(2^m)$, a third memory unit (R1) for storing n elements of the field $GF(2^m)$, at least one multiplying unit for multiplying the contents of said second memory unit by said one element of the field $GF(2^m)$ stored in said buffer memory, wherein the contents of said second memory unit is not obtained by an exchange with the contents of another memory means prior to effecting said multiplication operation, means for shifting either one of said first, second or third memory units and means for adding the contents of said second memory unit or a part of it to the contents of said first or third memory unit or a part thereof, or vice versa.

According to one embodiment of the invention an apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$ comprises dedicated hardware means for multiplying two elements of the field $GF(2^m)$. According to still another embodiment of the invention said dedicated hardware means further comprise means for performing adding operation.

According to one embodiment of the invention an apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$ comprises dedicated hardware means for squaring an element of the field $GF(2^m)$ over the polynomial basis.

Brief Description of the Drawings

In the drawings:

Fig. 1 shows in block diagram form a preferred method according to an embodiment of the invention for the calculation of modular multiplicative inverse of an element of the arithmetic field $GF((2^m)^n)$;

Fig. 2 shows the validity of a process for multiplying a plurality of elements of the field $GF(2^m)$ by an element of said field and further shows the utilization of said process. The relation among the various indices which concern the operation of certain registers R1 and [R2 when the operation $R1 = R1 + [R2$ is effected for a specified condition $i < j$ is shown in Fig. 2A. The relation among the various indices which concern the operation of said registers R1 and [R2, when the operation $R1 = R1 + [R2$ is effected for a specified condition $i > j$, is shown in Fig. 2B. The relation among the various indices which concern the operation of said registers R1 and [R2 when the operation $[R2 = [R2 + R1$ is effected is shown in Fig. 2C;

Fig. 3 shows dedicated means for multiplying two elements of the field $GF(2^8)$ over what is known as the polynomial basis;

Fig. 4 shows other dedicated means for multiplying two elements of the field $GF(2^8)$ over the polynomial basis;

Fig. 5 shows dedicated means for multiplying two elements of the field $GF(2^8)$ over what is known as the normal basis;

Figs. 6A and 6B show in block diagram form two variations of an apparatus according to an embodiment of the invention which comprises means for multiplying two elements of the field $GF(2^m)$ and means for performing an addition operation, both of said means being integrated within one unit. The case of effecting such a multiplication and addition without the use of a pre-processing unit is shown in Fig. 6A. The case of effecting such a multiplication and addition with the use of a pre-processing unit is shown in Fig. 6B;

Fig. 7 shows an apparatus for multiplying a plurality of elements of the field $GF(2^m)$ by an element of said field according to an embodiment of the

invention. Said apparatus, which uses one hardware multiplier of two elements of the field $GF(2^m)$, is shown in Fig. 7A. Said apparatus, which uses a plurality of hardware multipliers without the use of a pre-processing unit is shown in Fig. 7B. Said apparatus, which uses a plurality of hardware multipliers and which further uses a pre-processing unit, is shown in Fig. 7C;

Fig. 8A shows the use of a dedicated unit which according to an embodiment of the invention squares an element of the field $GF(2^m)$. Fig. 8B shows one implementation of said dedicated unit, which squares an element of the field $GF(2^8)$;

Fig. 9A shows a method of calculating the modular multiplicative inverse of an internal field $GF((2^5)^3)$, which replaces the field $GF(2^m)$. One preferred apparatus for effecting the shift operations executed in the implementation of said method is shown in Fig. 9B. Fig. 9C shows a preferred apparatus for effecting the multiplication operations executed in the implementation of said method.

Detailed Description of Preferred Embodiments

The invention provides an improved method for calculating multiplicative inverses over the field $GF((2^m)^n)$, a method in which the number of moving operations among various memory means is significantly reduced and the control over said operations is significantly simplified in comparison to the existing prior art methods. The invention further provides an apparatus for calculating multiplicative inverses over the field $GF((2^m)^n)$ according to said method and means of integrating the three fundamental operations effected in the implementation of Elliptic Curve Cryptography over $GF((2^m)^n)$ (the calculation of the multiplicative inverse of an element of the field $GF((2^m)^n)$, the calculation of the product of two elements of said field and the squaring of an element of said field) into one efficient process, which can be carried out by software or hardware means.

At the fundamental algorithmic level, the calculation of modular multiplicative inverses over the field $GF((2^m)^n)$ is effected by having two memory units, a first one for initially storing the generating polynomial of said external field and the second one for initially storing the field element to be inverted. Said two memory units are used to reduce the contents of said second memory unit by cancelling elements both from the most significant location and from the least significant location. This is done by reducing the degree of the polynomial stored in said second memory unit such that whenever zeros are generated by default, or exist at the least significant location, these zeros are cancelled as well, thereby reducing the overall number of the executed arithmetic operations.

A particular feature of the invention is the ability to effect a multiplication operation of a plurality of elements of the field $GF(2^m)$ by one element of said field, by always multiplying the contents of one specific memory unit by one element of said field, while the size of said one specific memory unit is fixed to store $n+1$ elements of said field. Furthermore, no exchange of contents between said one specific memory unit and other memory units precedes said multiplication. In particular, the final result is generated in a location designated at the beginning of the process.

All the modular arithmetic operations which are needed to be performed over the generating polynomial $g(x)$ of the external field of $GF((2^m)^n)$ concern only the contents of said one specific memory unit.

The method of the invention is valid for the case in which m and n are relatively prime and where the generating polynomial $g(x)$ of the external field of $GF((2^m)^n)$ has binary coefficients. Of course, the generating polynomial $f(x)$ of the internal field $GF(2^m)$ has binary coefficients as well.

A left-shift of any memory unit in all explanations hereinafter refers to performing a division operation over the field $GF((2^m)^n)$, as will be clear to those skilled in the art. If for any technical purpose a division operation is effected by a right-shift operation, then the direction of the indicated shifts (left or right) should be inverted.

The method of the invention comprises having memory units R_0 , R_1 and R_2 , wherein R_0 and R_2 are capable of storing $n+1$ elements of the field $GF(2^m)$ and R_1 is capable of storing n elements of said field. While said memory unit R_1 physically stores n elements, the index of the left-most element stored in R_1 is denoted as i .

As was defined, R_{ij} denotes an element of the field $GF(2^m)$ stored in the j -th place in memory unit R_i , wherein the index of the first (left) element is 0.

Hereinafter, h denotes the location within said memory unit R_0 of the right-most non-zero element. The value of this non-zero element is 1 (as an m -bit value) throughout the entire process.

Said memory unit R_2 consists of three sections. The left section, denoted as $R_2]$, becomes shorter during the process. k denotes the index of the last element of $R_2]$ within R_2 , where the index of the first element is 0. The right section of R_2 , denoted as $[R_2$, lengthens during the process. j denotes the index of the first element of $[R_2$ within R_2 wherein the index of the last element is n . The middle section of R_2 , which can be of length 0, is irrelevant to the process and contains 0s in practice.

The function of the method of the invention is shown in Fig. 1 and is better understood from observing the following Pseudo-code 3, which executes the same process. Comments under Pseudo-code 3 and the explanation which follows it further clarify the method of the invention. All the shifts indicated in the process are left shifts. (If the indexing is from the right, which is not the case in the following descriptions, then all shifts are to the right). The left element in a shifted memory unit is shifted out and discarded.

$R_{00} * R_{20}^{-1}$ denotes an element of the field $GF(2^m)$ obtained by multiplying the elements R_{00} and R_{20}^{-1} in which R_{ij}^{-1} denotes the modular multiplicative inverse over said field of the element R_{ij} . $R_k * Y$ means multiplying each element of the field $GF(2^m)$ stored in memory unit R_k by the field element Y .

The method of the invention suits the case in which the polynomial $g(x)$ is a trinomial in which s is the index of the middle non-zero coefficient. Extensions to other forms of $g(x)$ will be clear to those skilled in the art.

Pseudo-code 3. Calculating the multiplicative inverse of an element of the field $GF((2^m)^n)$

Initially: (as indicated in 101 in Fig. 1)

h, i and j are set to n ; k is set to $n-1$; p is set to 0.

$R0$ contains the coefficients of the trinomial $g(x)$ which are 1 elements (as elements of the field $GF(2^m)$) at locations 0, s and n ;

$R2]$ contains the element $b(x)$ of the field $GF((2^m)^n)$ which is to be inverted;

[$R2$ contains a 1.

1 If $R2_k = 0$ then $k = k-1$ and go to 1 (as indicated in 102 and 103 in Fig. 1)

(comment: the above loop decreases the value of k according to the number of 0's on the right of $R2]$.)

2 If $k = 0$ go to 4 (as indicated in 104 in Fig. 1)

(comment: currently, $R2]$ consists of a single non-zero element of the field $GF(2^m)$.)

If $R2_0$ not equal 0 go to 3 (as indicated in 111 in Fig. 1)

shift $R2]$ $k = k - 1$ $i = i + 1$ $p = p + 1$ go to 2

(as indicated in 113 in Fig. 1)

(comment: the above loop shifts-left $R2]$ until the left cell contains a non-zero element. [$R2$ does not shift. Instead of this, p is updated.)

-15-

3 If $R00 \neq 0$ then $Y = R00 \cdot R20^{-1}$ (as indicated in 112 and 119 in Fig. 1)
and

$R2 = R2 \cdot Y$ $R0 = R0 + R2$ $R1 = R1 + [R2$

if $j < i$ then $i = j$

(as indicated in 120 in Fig. 1)

(comment: After the above is executed, $R00 = 0$. All $+$ notations mean an XOR operation. By definition, the addition of $[R2$ to $R1$ is from index j to index n in $R1$. i denotes, as defined, the index of the left-most element stored in $R1$.)

shift $R0$ $i = i - 1$ $h = h - 1$ (as indicated in 114 in Fig. 1)

If $R2h = 0$ go to 3 (as indicated in 115 in Fig. 1)

(comment: The above loop shifts $R0$ and $R1$ to the left, while taking care that the element of the field $GF(2^m)$ stored on the left of $R0$ is 0; the shift operation continues until the h -th element of $R0$, which always contains the value 1, is positioned across the k -th element of $R2$], where the next operations convert this element into 0. This way, $R2$ becomes shorter.)

$Y = R2k^{-1}$ (as indicated in 116 in Fig. 1)

$R2 = R2 \cdot Y$ $R2] = R2] + R0$ $[R2 = [R2 + R1$ (as indicated in 117 in Fig. 1)

$j = i$ $k = k - 1$ go to 1 (as indicated in 118 in Fig. 1)

(comment: After the above is executed, $R2]k = 0$. By definition, the addition of $R1$ to $[R2$ is from index i to index n in $R2$.)

4 $Y = R20^{-1}$ (as indicated in 105 in Fig. 1)

$R2 = R2 \cdot Y$ shift $R2$ j times (as indicated in 106 in Fig. 1)

(comment: the last shift puts $[R2$ at the left side of the memory unit $R2$.)

$r = n + p - j - 1$ (as indicated in 107 in Fig. 1)

for $i = 0$ to r

$R2n = R20$ $R2s = R2s + R20$ shift $R2$

(as indicated in 108 and 109 in Fig. 1)

stop The final content of R2, starting with index 0, is $b^{-1}(x)$
 (as indicated in 110 in Fig. 1).

Explanation:

R0 initially contains the generating polynomial $g(x)$ of the field $GF((2^m)^n)$. The two memory units R0 and R2] always contain two polynomials which are obtained from $g(x)$ and $b(x)$ by multiplying both by the same element of the field $GF(2^m)$, or by divisions by x over the field $GF((2^m)^n)$, or by adding the content of one of said two memory units to the other. The GCD (Greatest Common Divisor) of the polynomials stored in these two memory units is 1, since $g(x)$ is irreducible. Said GCD remains unchanged until R2] contains a guaranteed single non-zero element of the field $GF(2^m)$, when the procedure then gets to the line indicated by 4. Corresponding operations are performed on [R2 and R1, whose initial contents are respectively 1 and 0. This is done in accordance with the Euclid algorithm and according to prior art, as will be clear to persons skilled in the art.

All the calculations up to line 4 in Pseudo-code 3 are intended to shorten R2], whose initial content consists of the element $b(x)$ of the field $GF((2^m)^n)$ and whose multiplicative inverse is to be calculated, the final contents of R2] before line 4 is executed being a single non-zero element of the field $GF(2^m)$.

R2] is shortened by cancelling, one at a time, the right (highest-degree) coefficient stored in R2] (whose index is k). Whenever the left element in R2] is 0, R2] is left-shifted and said 0 is cancelled. All other operations are executed, while the left element in R2] is non-zero. By definition, the initial value of the right element in R0 is 1 (as an element of the field $GF(2^m)$). Throughout the process, the value of this element never changes. This element 'slides' across R2], via shifts of R0 and cancels the right non-zero element in R2]. Said shifts of R0 are effected by first cancelling the lowest-degree coefficient stored in R0 and then left-shifting R0. Different tasks are thereby assigned to each of said three memory units R0, R1 and R2, in order to perform said shortening of R2].

-17-

A clear feature of Pseudo-code 3, according to an embodiment of the method of the invention, concerns cancellations of right non-zero elements in R2] while shifting R2] in the case in which there is a left 0 in R2], thereby reducing the contents of R2] from both sides. This feature is also applicable to the calculation of modular multiplicative inverses over the field $GF(2^n)$, which can be viewed as the field $GF((2^m)^n)$ for the case $m=0$, in which case the elements of the field $GF(2^m)$ are a single bit.

According to an embodiment of the method of the invention, said memory unit [R2 is not shifted. (p is the accumulated number of left-shifts of [R2 which should have taken place, where said p is used in the post-processing starting at line 4 in said Pseudo-code 3.) [R2, which is not shifted and which initially contains a single non-zero value, 'grows' towards its left by additions of R1's elements to [R2. It is observed that not shifting [R2 creates the following conditions regarding various operations indicated in Pseudo-code 3:

1. The relation among the various indices which concern the operation of said registers R1 and R2 when the operation $R1 = R1 + [R2$ is effected for $i < j$ is shown in Fig. 2A, clarifying a condition under which the last element of [R2, when added to a corresponding place in R1, does not fall on the right of the element with index $i+n-1$ in R1, thereby keeping the length of R1 as n.
2. The relation among the various indices which concern the operation of said registers R1 and R2, when the operation $R1 = R1 + [R2$ is effected for $i > j$, is shown in Fig. 2B, clarifying a condition under which said operation always takes place when the last (i-j) elements of R1 are 0. Therefore, while the index i of the first element of R1, after the addition, points at an element which is (i-j) places to the left of the first element of R1 before the addition, the length of R1 is still kept as n, since the right (i-j) 0s are deleted.
3. The relation among the various indices which concern the operation of said registers R1 and R2 when the operation $[R2 = [R2 + R1$ is effected is

-18-

shown in Fig. 2C. As clarified in the illustration, said operation always takes place under the condition where $k < i \leq j$. As observed in Pseudo-code 3, said operation $R2 = [R2 + R1$ is always associated by updating the value of j to equal i . The condition $i \leq j$ means that j gets lower or remains the same. This is the only way by which $[R2$ expands to the left, as $[R2$ never shifts. The condition $i > k$ guarantees that the sum of the lengths of $R2$ and $[R2$ does not exceed $n+1$. Fig. 2C also shows the existence of a condition under which said operation $R2 = [R2 + R1$ is effected when all the elements in $R1$ of index $> n$ are 0. Therefore, $R2$ never grows to the right beyond the index n .

The existence of said conditions specified in 1, 2 and 3 above is guaranteed by not shifting $[R2$, where the existence of said conditions clearly enables the following features in accordance with an embodiment of the invention:

any multiplication of a plurality of elements of the field $GF(2^m)$ by one element of said field concerns the multiplication of the content of said one memory unit $R2$ by said one element, as indicated by the operation $R2 = R2 * Y$ of Pseudo-code 3;

said one memory unit $R2$ is addressed as one unit which stores a fixed number of $n+1$ elements of the field $GF(2^m)$.

Another feature in accordance with an embodiment of the invention concerns the fact that no exchange of contents between said memory unit $R2$ and another memory unit precedes the multiplication of the plurality of elements of the field $GF(2^m)$ stored in said memory unit $R2$ by one element of said field.

The operations indicated in Fig. 1 and in said Pseudo-code 3 as

for $i = 0$ to r

$R2_n = R2_0$

$R2_s = R2_s + R2_0$

shift $R2$

can be implemented by operating said memory unit R2 as a feedback shift register using circuitry of the form, indicated, for example, in W.W. Peterson and E.J. Weldon, *Error Correcting Codes*, The MIT Press, 1972, pp. 170-205.

Embodiments of the invention hereinafter described concern the use of dedicated means for multiplying two elements of the field $GF(2^m)$. A way of effecting said dedicated means is shown in Fig. 3 for the case $m=8$ and for a generating polynomial $f(x) = 1 + x + x^6 + x^7 + x^8$, where the implementation is over what is known as the polynomial basis. The two multiplied field elements are $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ and $(b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7)$ yielding the field element $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$. The shown dedicated means execute the modular polynomial multiplication $(a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7) \cdot (b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7) \bmod (1 + x + x^6 + x^7 + x^8) = (c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 + c_5x^5 + c_6x^6 + c_7x^7)$. The unit 11 comprises eight 'AND' logic gates, which perform the multiplication $a_0 \cdot (b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7)$. Similarly, each of the other groups of eight 'AND' logic gates shown in Fig. 3 below unit 11 multiply the elements $a_1, a_2, a_3, a_4, a_5, a_6, a_7$ by the polynomial $(b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7)$. The unit 12 comprises five 'XOR' logic gates which perform the modular arithmetic operation over $(1 + x + x^6 + x^7 + x^8)$ in the generation of the coefficient c_0 of the result $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$. Similarly, each of the other groups of 'XOR' logic gates shown in Fig. 3 below unit 12 performs the modular arithmetic operation over $(1 + x + x^6 + x^7 + x^8)$ in the generation of other coefficients of the result $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$.

Fig. 4 shows another implementation of said dedicated means for multiplying two elements of the field $GF(2^m)$ for the case $m=8$ and for a generating polynomial $f(x) = 1 + x + x^6 + x^7 + x^8$ over the polynomial basis. The unit 20 is a pre-processing unit which operates on the input field element $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ and which performs in advance the modular arithmetic operations over $(1 + x + x^6 + x^7 + x^8)$. The output of unit 20 enters a post-processing unit 21, whose other input is the field element $(b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7)$. The output of unit 21 are the coefficients of the

field element $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$, which is the result of the operation $(a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7) * (b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7) \bmod (1 + x + x^6 + x^7 + x^8)$. The unit 21 comprises eight of the units 22, each one generating one of the coefficients $c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7$, where said unit 22 comprises eight 'AND' logic gates and a multiple input 'XOR' gate.

Fig. 5 shows another implementation of said dedicated means for multiplying two elements of the field $GF(2^m)$, for the case $m=8$ and for a generating polynomial $f(x) = 1 + x + x^6 + x^7 + x^8$ over what is known as the normal basis. Such an implementation is possible, as the eight roots of $f(x)$ are independent and the multiplication of the field elements $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ and $(b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7)$ of the field $GF(2^m)$ can therefore be effected, over the normal basis, generated by $f(x)$, by the operation $(a_0x + a_1x^2 + a_2x^4 + a_3x^8 + a_4x^{16} + a_5x^{32} + a_6x^{64} + a_7x^{128}) * (b_0x + b_1x^2 + b_2x^4 + b_3x^8 + b_4x^{16} + b_5x^{32} + b_6x^{64} + b_7x^{128}) \bmod (1 + x + x^6 + x^7 + x^8) = (c_0x + c_1x^2 + c_2x^4 + c_3x^8 + c_4x^{16} + c_5x^{32} + c_6x^{64} + c_7x^{128})$.

The unit 30 is a pre-processing unit which operates on the input field element $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ and performs all the modular arithmetic operations needed to be effected when executing operations over a normal basis. Unit 30 consists of seven units 32, each one receiving a cyclic shift of the eight input bits $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$, as will be clear to persons skilled in the art. The unit 30 further has a unit 33, which differs from said unit 32 by having one less 'XOR' gate. The output of unit 30 consists of eight groups, each consisting of eight bits. Each output group of unit 30 enters an array of eight 'AND' logic gates in the post-processing unit 31, while the other input to each of said 'AND' gates is a corresponding bit from $(b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7)$. Unit 31 consists of eight units, each of which is the same as said unit 22 of Fig. 4. The output of unit 31 is the field element $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$, which is the result of the multiplication of the field element $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ by the field element $(b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7)$.

Figs. 3, 4 and 5, which show dedicated means for multiplying elements of the field $GF(2^8)$ generated by $f(x) = 1 + x + x^6 + x^7 + x^8$, are given for illustration purposes. Similar circuits can be constructed by those skilled in the art for other values of m and for other generating polynomials $f(x)$.

An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, in which two elements of the field $GF(2^m)$ are multiplied by dedicated hardware means, forms an embodiment of the invention.

Pseudo-code 3 includes the multiplication operation $R2 = R2 * Y$, followed by addition operations of the general form $R2 = R2 + RI$ or $RI = RI + R2$. Fig. 6A shows a circuit according to an embodiment of the invention which integrates said multiplication and addition operations. Unit 40, framed by a broken line, comprises unit 41 followed by unit 42 for summing the result of unit 41 with the value RI_j . Unit 41 multiplies the values of $R2_j$ and Y as two elements of the field $GF(2^m)$ and can be implemented by one of the circuits of Fig. 3, 4 or 5 and their generalization to other values of m .

Fig. 6B shows in block diagram form a circuit according to an embodiment of the invention in which the multiplication and addition operations are integrated. Unit 45 is a pre-processor, such as unit 20 of Fig. 4 or unit 30 of Fig. 5 and can be generalized for different values of m . Unit 44 comprises a post-processor such as unit 21 of Fig. 4 (when the pre-processor is of unit 20 type), or such as unit 31 of Fig. 5 (when the pre-processor is of unit 30 type) and may be generalized for different values of m . Unit 43, framed by broken lines, is comprised of said unit 44, followed by summing unit 42, for summing the result of unit 44 with the value RI_j .

As indicated hereinbefore, one part of the inventive method for calculating a modular multiplicative inverse of an element of the field $GF((2^m)^n)$ concerns an ability to perform all multiplications of a plurality of elements of the field $GF(2^m)$ by one element of said field, by means of always storing said plurality of elements in one memory unit $R2$, addressed as one unit of a

fixed length. One embodiment of the invention is shown in Fig. 7A, which comprises a unit 51 for effecting multiplication of each element stored in R2 by one element which is stored in a buffer memory 50. Said unit 51 can be implemented, for example, by any of the circuits shown in Fig. 3, 4 or 5.

According to another embodiment of the invention, the multiplication of the plurality of elements of the field $GF(2^m)$, which are stored in memory unit R2, by one element of said field, is executed by applying a plurality of dedicated means, each one of the form of said unit 51, for effecting a simultaneous multiplication of said plurality of elements by said one element, where said one element is stored in said buffer memory 50. Such implementation is shown in Fig. 7B.

Fig. 7C shows another embodiment of the invention for effecting, in a slightly different way from that of Fig. 7B, a simultaneous multiplication of a plurality of elements by one element. In order to multiply one element of the field $GF(2^m)$ by the plurality of field elements stored in said memory unit R2, said one element, stored in buffer memory 50, forms the input to the pre-processing unit 52, where said pre-processing unit may comprise a unit such as unit 20 of Fig. 4 (when operating over the polynomial basis) or an alternative unit such as unit 30 of Fig. 5 (when operating over the normal basis). Each of the plurality of post-processing units 53 has two inputs, a first input for receiving a $GF(2^m)$ element from memory unit R2 and a second input for receiving the output from pre-processing unit 52. If the pre-processing unit 52 is of the form of unit 20 (Fig. 4) then each of the post-processing units 53 has the form of unit 21 of Fig. 4. If, however, the pre-processing unit 52 has the form of unit 30 (Fig. 5), then each post-processing unit 53 should have the form of unit 31 of Fig. 5.

Fig. 8A shows an apparatus for effecting the method of squaring an element of the field $GF((2^m)^n)$ stored in memory unit R1, hereinbefore described in Pseudo Code 2, according to one embodiment of the invention. Unit 60 effects the operation $R1i = [R1i]^2$ specified in said Pseudo Code. When operating over the polynomial basis of the field $GF(2^m)$, the operation of said unit 60 is

effected by a dedicated circuit, which utilizes the fact that squaring an element of the field of $GF(2^m)$, when operating over the polynomial basis, can be implemented as a vector-matrix multiplication, as will be clear to persons skilled in the art. Fig. 8B shows a design of a dedicated circuit which effects said vector-matrix multiplication for a generating polynomial $f(x) = 1 + x + x^6 + x^7 + x^8$.

An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, in which an element of the field $GF(2^m)$ is squared, over the polynomial basis, by dedicated hardware means, forms an embodiment of the invention.

An embodiment of the invention concerns an apparatus which composes into one operator the calculation of the multiplicative inverse of an element of the field $GF((2^m)^n)$, the calculation of the product of two elements of said field and the squaring of an element of said field. Said apparatus preferably comprises three memory units R0, R1 and R2, which store respectively, $n+1$, n and $n+1$ elements of the field $GF(2^m)$, where said memory units are effected by the following operations:

A first operation, in which all the elements of the field $GF(2^m)$ stored in said memory unit R2 are multiplied by one element of the same field stored in a buffer memory, where no such multiplication is effected in connection with the said memory units R0 or R1 and where no exchange of contents between said memory unit R2 and either one of said memory units R0 or R1 takes place prior to said multiplication;

A second operation, in which a variable number of elements of the field $GF(2^m)$ stored in said memory unit R2 are added, element by element, to elements stored in said memory unit R0;

A third operation, in which a variable number of elements of the field $GF(2^m)$ stored in said memory unit R0 are added, element by element, to elements stored in said memory unit R2;

A fourth operation, in which a variable number of elements of the field $GF(2^m)$ stored in said memory unit R2 are added, element by element, to elements stored in said memory unit R1;

A fifth operation, in which a variable number of elements of the field $GF(2^m)$ stored in said memory unit R1 are added, element by element, to elements stored in said memory unit R2;

A sixth operation, in which the elements of said memory unit R0 are left-shifted, either physically or by manipulations with indices;

A seventh operation, in which a variable number of the elements of said memory unit R2 are left-shifted, either physically or by manipulations with indices;

An eighth operation, in which the elements of said memory unit R2 are either left-shifted or right-shifted, either physically, or by manipulations with indices, while modular operations over the generating polynomial $g(x)$ of the field $GF((2^m)^n)$ are being performed;

A ninth operation, in which a variable number of elements of said memory unit R1 are right-shifted, either physically or by manipulations with indices.

It will now be shown how the aforesaid nine operations effect all the operations indicated in Pseudo-Codes 1, 2 and 3, concerning said memory units R0, R1 and R2, thereby forming a general operator which effects the three fundamental operations associated with Elliptic Curve Cryptographic applications, where said three operations are: a) calculating the product of two elements of the field $GF((2^m)^n)$, b) squaring of an element of said field, and c) calculating the multiplicative inverse of an element of said field.

Said nine operations, or a part of them, effect the operation of calculating the product of two elements of the field $GF((2^m)^n)$, indicated in said Pseudo-Code 1, as follows: The operation indicated in said Pseudo-Code 1 by $R2 = R2 * Y$ is effected by said first operation. The operation $R1 = R1 + R2 * R0i$ of Pseudo-Code 1 is preferably effected by said first and fourth operations, or by their integration, as shown in Fig. 6. The operations shift-right R2, $R2_0 = R2_n$ and $R2_s = R2_s + R2_n$ of Pseudo-Code 1 are effected by said eighth operation.

Said nine operations, or a part of them, effect also the operation of squaring an element of the field $GF((2^m)^n)$, indicated in said Pseudo-Code 2, as follows: The operation $R2 = R2 + R1$ of Pseudo-Code 2 is effected by the said fifth operation. The operations shift-right $R2$, $R2_0 = R2_n$ and $R2_s = R2_s + R2_n$ of Pseudo-Code 2 are effected by the said eighth operation. The operation shift-right $R1$ from index $(2i-1)$ of Pseudo-Code 2 is effected by the said ninth operation.

Said nine operations, or a part of them, effect the operation of calculating the modular multiplicative inverse of an element of the field $GF((2^m)^n)$, indicated in said Pseudo-Code 3, as follows: The operation indicated in said Pseudo-Code 3 by $R2 = R2 * Y$ is effected by said first operation. The operation indicated in said Pseudo-Code 3 by $R0 = R0 + R2$ is effected by said second operation. The operation indicated in said Pseudo-Code 3 by $R2 = R2 + R0$ is effected by said third operation. The operation indicated in said Pseudo-Code 3 by $R1 = R1 + R2$ is effected by said fourth operation. The operation indicated in said Pseudo-Code 3 by $R2 = R2 + R1$ is effected by said fifth operation. The operations indicated in said Pseudo-Code 3 by shift $R0$ and shift $R2$ are respectively effected by said sixth and seventh operations. The operations $R2_n = R2_0$, $R2_s = R2_s + R2_0$ and shift (left) $R2$ of Pseudo-Code 3 are effected by said eighth operation.

According to still another embodiment of the invention the internal sub-field $GF(2^m)$ can be replaced by a sub-field of 2^r elements, which in itself has a sub-field. The calculation of modular multiplicative inverses over a sub-field of 2^r elements can be efficiently performed by the application of a dedicated hardware which processes p -bit values at a time, for p being a divisor of r .

Fig. 9A shows a method for calculating modular multiplicative inverses over an internal sub-field for the case $r = 15$ and $p = 5$. That is, the internal sub-field $GF(2^m)$ is replaced here by the field $GF((2^5)^3)$. The functioning of the method of Fig. 9A is better understood from the following Pseudo-Code 4, which executes the same process. The comments of Pseudo-Code 4 further clarify the method.

Figs. 9B and 9C show a preferred apparatus for effecting some of the operations indicated in the method shown in Fig. 9A and the following Pseudo-Code 4. The four registers R0, R1, R2 and R3 of Fig. 9B act on 5-bit values. That is, 5-bits are shifted at a time and are added or multiplied at a time. The thick lines in Fig. 9B represent 5-bit lines. The thick XOR gate operates on 5-bit values. R_{ij} denotes the 5 bits stored in location j of register i . The four flags, flag0, flag1, flag2 and flag3, respectively indicate the existence of conditions under which R20, R21, R22 and R00 contain zeros. It should be noted that the value of the flag is 1 when the corresponding five bits are all 0s.

Fig. 9C shows a possible apparatus to be used for effecting the multiplication operations of Fig. 9A. These operations concern multiplying the contents of R3, or the contents of R2-R3 (altogether, six values of 5-bit each), by a 5-bit value Z. The unit marked as 70 is a 5-bit inverter which yields the inverse of the element Y modulo the generating polynomial of the sub-field $GF(2^5)$. The unit marked as 71 multiplies two elements of the sub-field $GF(2^5)$, one of which is Y^{-1} and the other is R00. The unit marked as 72 is a multiplexor whose output is Z, while $Z = Y^{-1}$ if sel = 0 and $Z = R00 * Y^{-1}$ if sel = 1 (sel indicates the select input to the multiplexor). Unit 73 multiplies the contents of R3, or the contents of R2-R3, by the value Z.

Pseudo-Code 4: A method for calculating modular multiplicative inverses over the field $GF((2^5)^2)$

Initially: R00 contains a 1, R01 contains a 1, R02 contains a 0

(As indicated in 301 in Fig. 9A.

comment: said 1, 1, 0 are stored as elements of the field $GF(2^5)$;

R1 contains zeros;

R2 contains the element $b(x)$ to be inverted;

R30 contains a 1, R31 and R32 contain a 0

(*comment:* said 1, 0, 0 are stored as elements of the field $GF(2^5)$. R0 stores the coefficients of the generating polynomial $1+x+x^3$ of the external field, while the coefficients of x^3 are not stored in practice.)

if flag0 is not equal to 1, go to 1 (as indicated in 302 in Fig. 9A)

(comment: if flag0=1 then the contents of R2 are of the form 0XX where X denotes a general value, but the two Xs cannot be both 0. Otherwise, the contents of R2 are of the form NXX, wherein N denotes a non-zero value. The notations X and N are also used later.)

shift R2 and R3 (as indicated in 303 in Fig. 9A)

(comment: here the contents 0XX of R2 are left-shifted, yielding XX0.)

if flag0 not equal 1 go to 1 (as indicated in 304 in Fig. 9A)

(comment: if flag0 = 1, then the contents of R2 are of the form 0X0. Otherwise, the content of R2 are NX0.)

shift R2-R3 and go to 4 (as indicated in 325 in Fig. 9A)

(comment: the current contents of R2 are N00.)

1 if flag1=flag2=1, go to 4 (as indicated in 305 in Fig. 9A)

(comment: if flag1= flag2 = 1, then the current contents of R2 are N00. Otherwise, the current contents of R2 are NXX.)

set Y = R20 sel=0 (as indicated in 306 in Fig. 9A)

R2-R3 = (R2-R3)*Z (as indicated in 307 in Fig. 9A)

(comment: the current contents of R2 are 1XX.)

R0-R1 = (R0-R1)+(R2-R3) (as indicated in 308 in Fig. 9A)

(comment: the current contents of R0 are 0XX.)

shift R0-R1 set R0z = 1 (as indicated in 309 in Fig. 9A)

(comment: the current contents of R0 are XX1.)

if flag2=1 go to 2 (as indicated in 310 in Fig. 9A)

(comment: if flag2=1, then the contents of R2 are 1N0. Otherwise, the contents of R2 are 1XN.)

set Y = R2₂ sel=0 (as indicated in 311 in Fig. 9A)

R2-R3 = (R2-R3)*Z (as indicated in 312 in Fig. 9A)

(comment: the current contents of R2 are NX1.)

R2-R3 = (R2-R3)+(R0-R1) (as indicated in 313 in Fig. 9A)

(comment: the current contents of R2 are XX0.)

if flag0=1, shift R2-R3 and go to 4 (as indicated in 314 and 316 in Fig. 9A)

(comment: if flag0=1, then the contents of R2 are 0N0 and the shift yields the contents N00. If flag0=0 then the contents of R2 are NX0.)

if flag1=1, go to 4 (as indicated in 315 in Fig. 9A)

(comment: if flag1=1, then the contents of R2 are N00. If flag1=0, then the contents of R2 are NN0.)

2 if flag3=1, go to 3 (as indicated in 317 in Fig. 9A)

(comment: if flag3=1, then the contents of R0 are 0X1. If flag3=0, then the contents of R0 are NN1.)

set Y = R2₀ sel=1 (as indicated in 318 in Fig. 9A)

R2-R3 = (R2-R3)*Z (as indicated in 319 in Fig. 9A)

(comment: the current contents of R2 are R00X0.)

R0-R1 = (R0-R1)+(R2-R3) (as indicated in 320 in Fig. 9A)

(comment: the current contents of R0 are 0X1.)

3 shift R0-R1 (as indicated in **321** in Fig. 9A)

-30-

CLAIMS

1. A method for calculating modular multiplicative inverses of a Galois Field $GF((2^m)^n)$ comprising the steps of:

- having first (R0), second (R2) and third (R1) memory units, storing respectively, $n+1$, $n+1$ and n elements of the field $GF(2^m)$;
- storing in said first memory unit the generating polynomial ($g(x)$) of the external field of said Galois Field;
- storing in said second memory unit the field element to be inverted, followed with a 0;
- adding the contents of the first part of said second memory unit (R2]) to the contents of said first memory unit and adding the contents of the second part of said second memory unit ([R2) to the contents of said third memory unit in the lines of the Euclid algorithm;
- adding the contents of a part of said first memory unit to the contents of the first part of said second memory unit (R2]) and adding the contents of a part of said third memory unit to the contents of the second part of said second memory unit ([R2) in the lines of the Euclid algorithm and in order to convert into a zero value the element of the field $GF(2^m)$ stored at the most significant place at said first part of said second memory unit;
- effecting shift operations on said first memory unit and said first part of said second memory unit, while not shifting said second part of said second memory unit, in order to cancel elements of the field $GF(2^m)$ stored at the least significant place at said first and second memory units;
- effecting multiplication operations in which the entire contents of said second memory unit are multiplied by one element of said field in the lines of the Euclid algorithm;

whereby:

a. to reduce the contents of said first part of said second memory unit in the lines of the Euclid algorithm by canceling elements both from the most significant location and from the least significant location of said first part of said second memory unit; and

b. to calculate modular multiplicative inverses by effecting steps in which a multiplication of a plurality of elements of the field $GF(2^m)$ by one element of said field is always effected when said plurality of elements is stored in one specific memory unit (R2) of a fixed length, without a prior exchange of contents between said one specific memory unit and another memory unit.

2. The method of claim 1, wherein each one of said elements of the field $GF(2^m)$ is a single bit.

3. A method for effecting the operations of calculating the product of two elements of the field $GF((2^m)^n)$, calculating the square of an element of said field and calculating the multiplicative inverse of an element of said field, comprising the steps of:

- having first (R0), second (R2) and third (R1) memory units for storing respectively, $n+1$, $n+1$ and n elements of the field $GF(2^m)$;
- multiplying all the elements of the field $GF(2^m)$ stored in said second memory unit by one element of said field, where no such multiplication is effected in connection with said first or third memory units and where no exchange of contents between said second memory unit and said first or third memory units takes place prior to said multiplication;
- adding a variable number of elements of the field $GF(2^m)$ stored in said first memory unit, to elements stored in said second memory unit;
- adding a variable number of elements of the field $GF(2^m)$ stored in said second memory unit, to elements stored in said first memory unit;
- adding a variable number of elements of the field $GF(2^m)$ stored in said third memory unit, to elements stored in said second memory unit;
- adding a variable number of elements of the field $GF(2^m)$ stored in said second memory unit, to elements stored in said third memory unit;
- shifting, either physically or by manipulations with indices, the elements stored in said first memory unit;
- shifting, either physically or by manipulations with indices, a variable number of elements stored in said second memory unit;

- shifting, either physically or by manipulations with indices, the elements stored in said second memory unit, while modular operations over the generating polynomial of the field $GF((2^m)^n)$ are being performed; and
- shifting, either physically or by manipulations with indices, a variable number of elements stored in said third memory unit;

whereby to have a unified method which effects, by effecting any of the aforesaid steps, the operations of calculating the product of two elements of the field $GF((2^m)^n)$, calculating the square of an element of said field and calculating the multiplicative inverse of an element of said field.

4. A method for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, wherein said value m equals the product of two different numbers, p and r and wherein the calculation of modular multiplicative inverses over the internal field $GF(2^m)$ is performed by processing p -bit values at a time.

5. An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, comprising:

- a buffer memory for storing one element of the field $GF(2^m)$.
- a first memory unit ($R0$) for storing $n+1$ elements of the field $GF(2^m)$.
- a second memory unit ($R2$) for storing $n+1$ elements of the field $GF(2^m)$.
- a third memory unit ($R1$) for storing n elements of the field $GF(2^m)$.
- at least one multiplying unit for multiplying the contents of said second memory unit by one element of the field $GF(2^m)$ stored in said buffer memory, wherein the content of said second memory unit is not obtained by an exchange with the content of another memory means prior to effecting said multiplication operation.
- means for shifting either one of said first, second or third memory units.
- means for adding the contents of said second memory unit or a part of it to the contents of said first or third memory unit or a part thereof, or vice versa.

6. An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, comprising dedicated hardware means for multiplying two elements of the field $GF(2^m)$.
7. An apparatus according to claim 6, wherein said dedicated hardware means include means for performing an adding operation.
8. An apparatus for carrying out the method of claim 1, in which a multiplication of a plurality of elements of the field $GF(2^m)$ stored in said specific memory unit (R2) by one element of said field stored in a buffer memory is effected by using a plurality of circuits, each such circuit multiplying said one element by an element stored in said specific memory unit.
9. An apparatus for carrying out the method of claim 1, in which a multiplication of a plurality of elements of the field $GF(2^m)$ stored in said specific memory unit by one element of said field stored in a buffer memory is effected by using one pre-processing circuit which processes said one element and a plurality of post-processing circuits, each such circuit operating on two values, one being the output of said pre-processing circuit and the other value is an element stored in said specific memory unit.
10. An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, comprising dedicated hardware means for squaring an element of the field $GF(2^m)$ over the polynomial basis.
11. An apparatus for effecting Elliptic Curve Cryptographic operations over the field $GF((2^m)^n)$, essentially as described and with particular reference to the drawings.

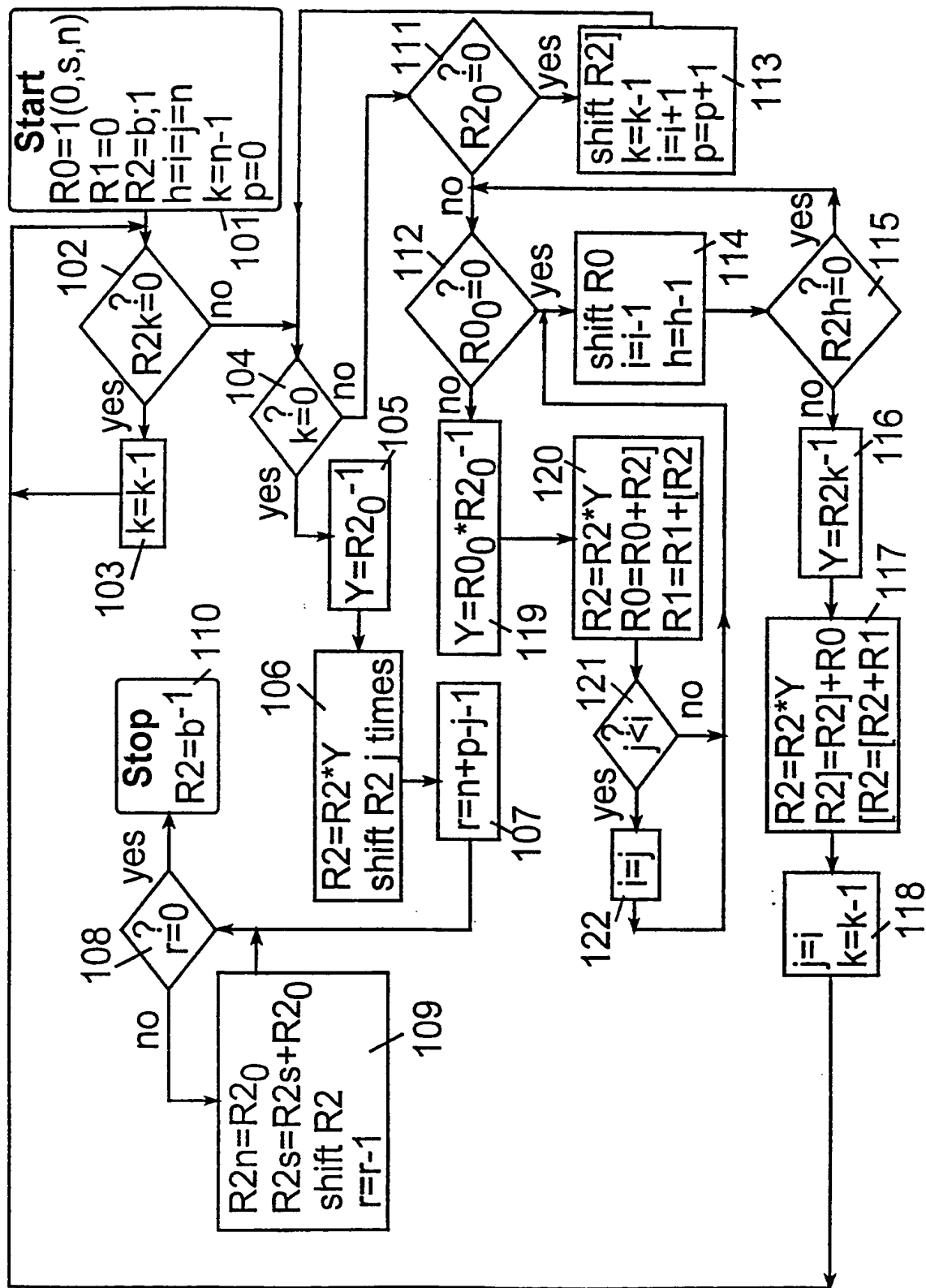
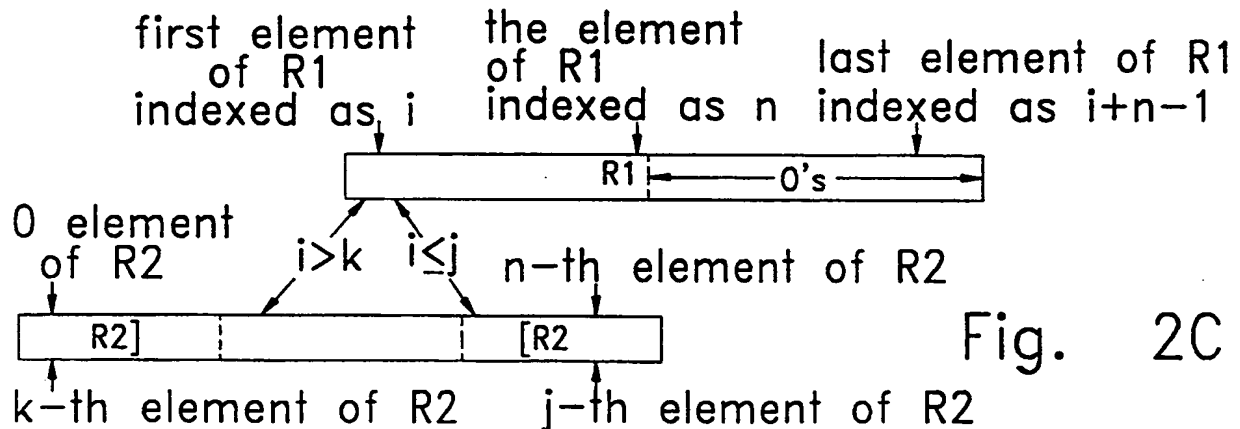
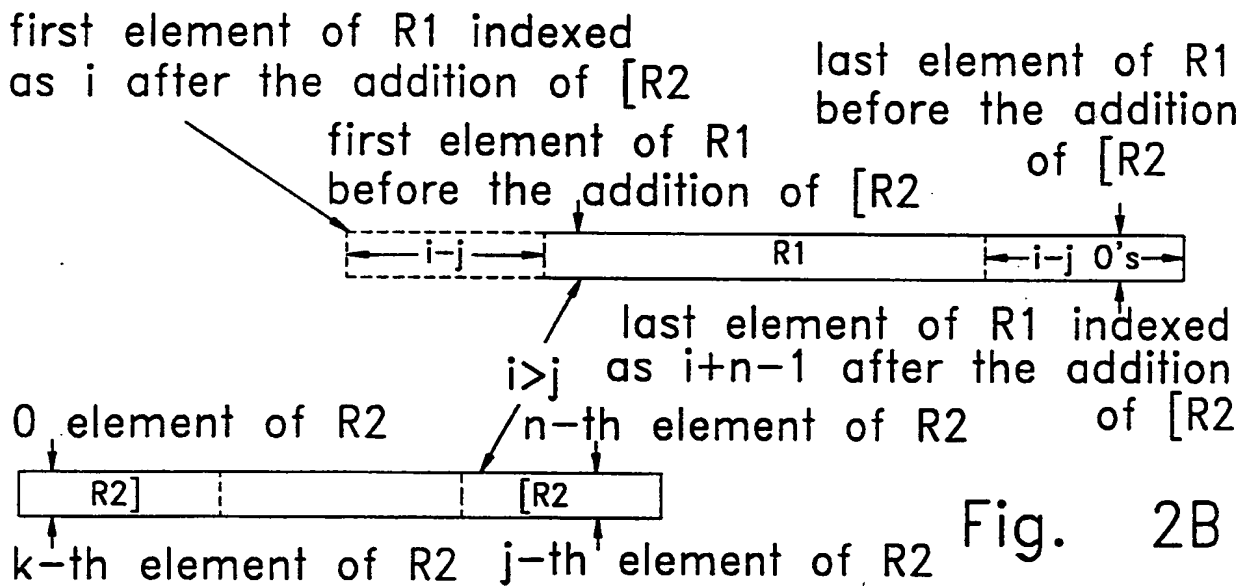
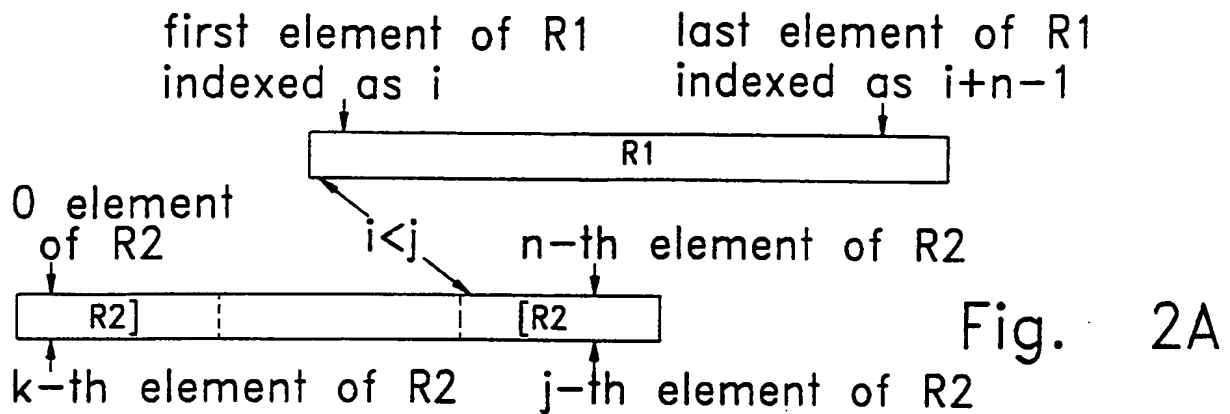


Fig. 1

2/10



3/10

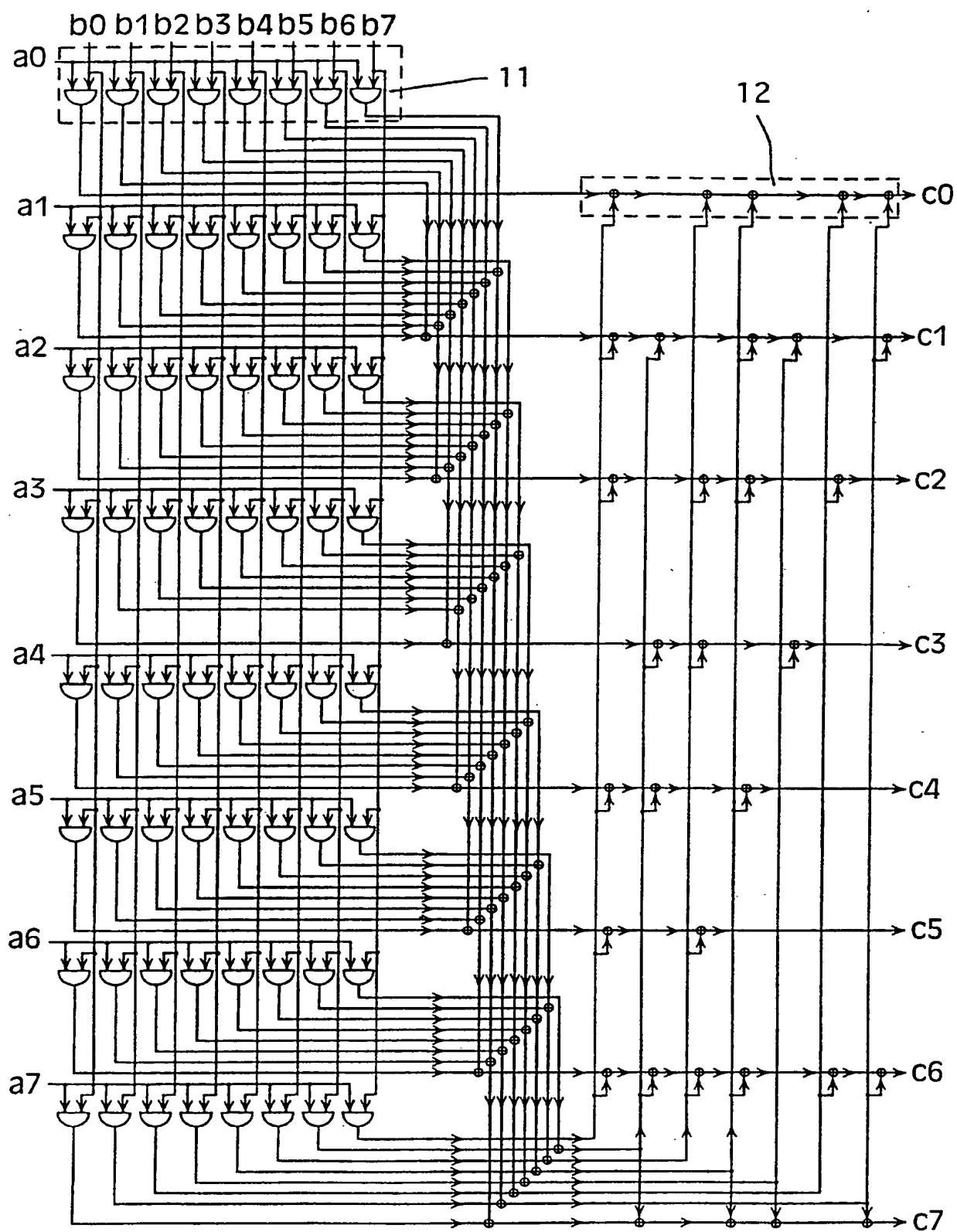


Fig. 3

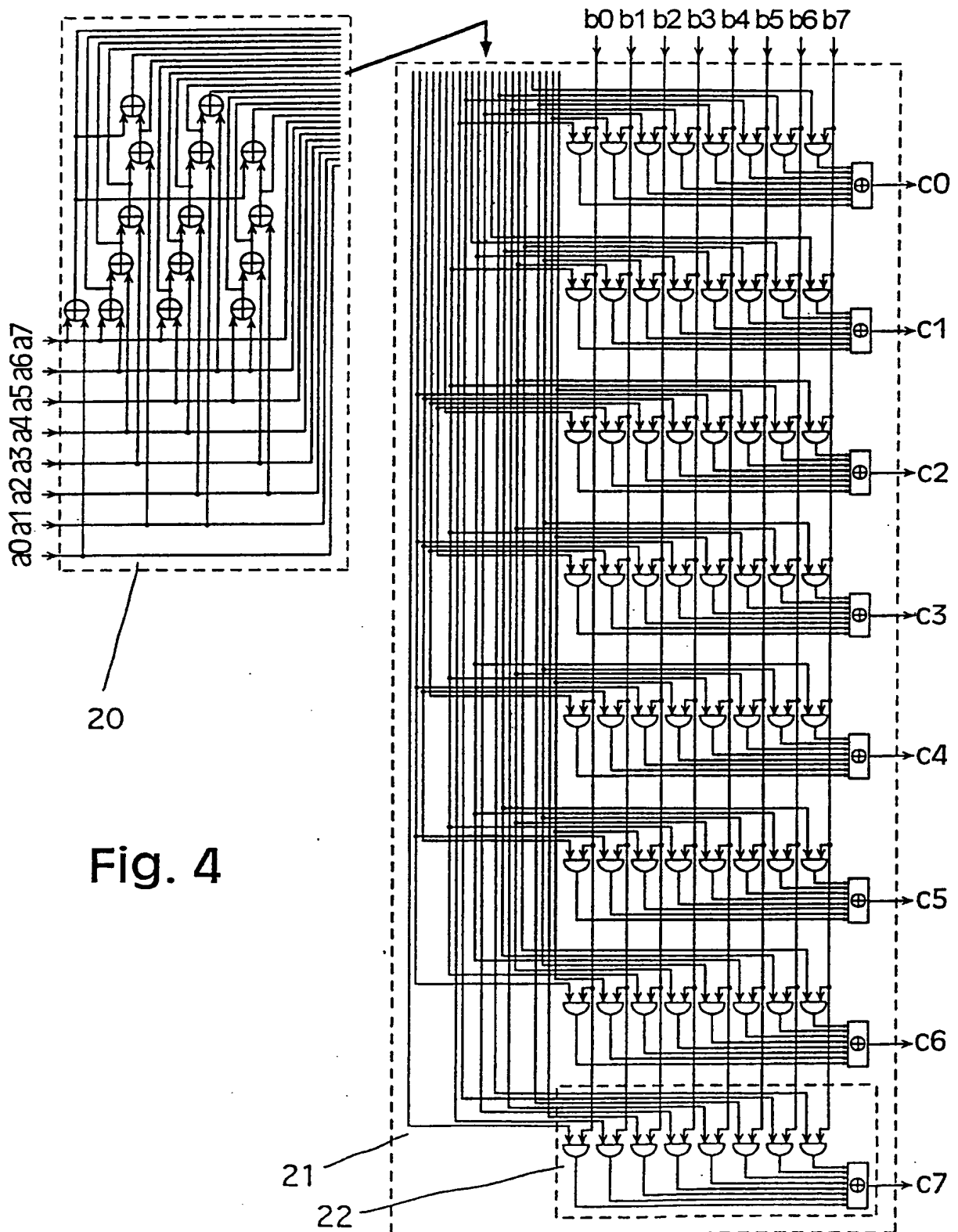
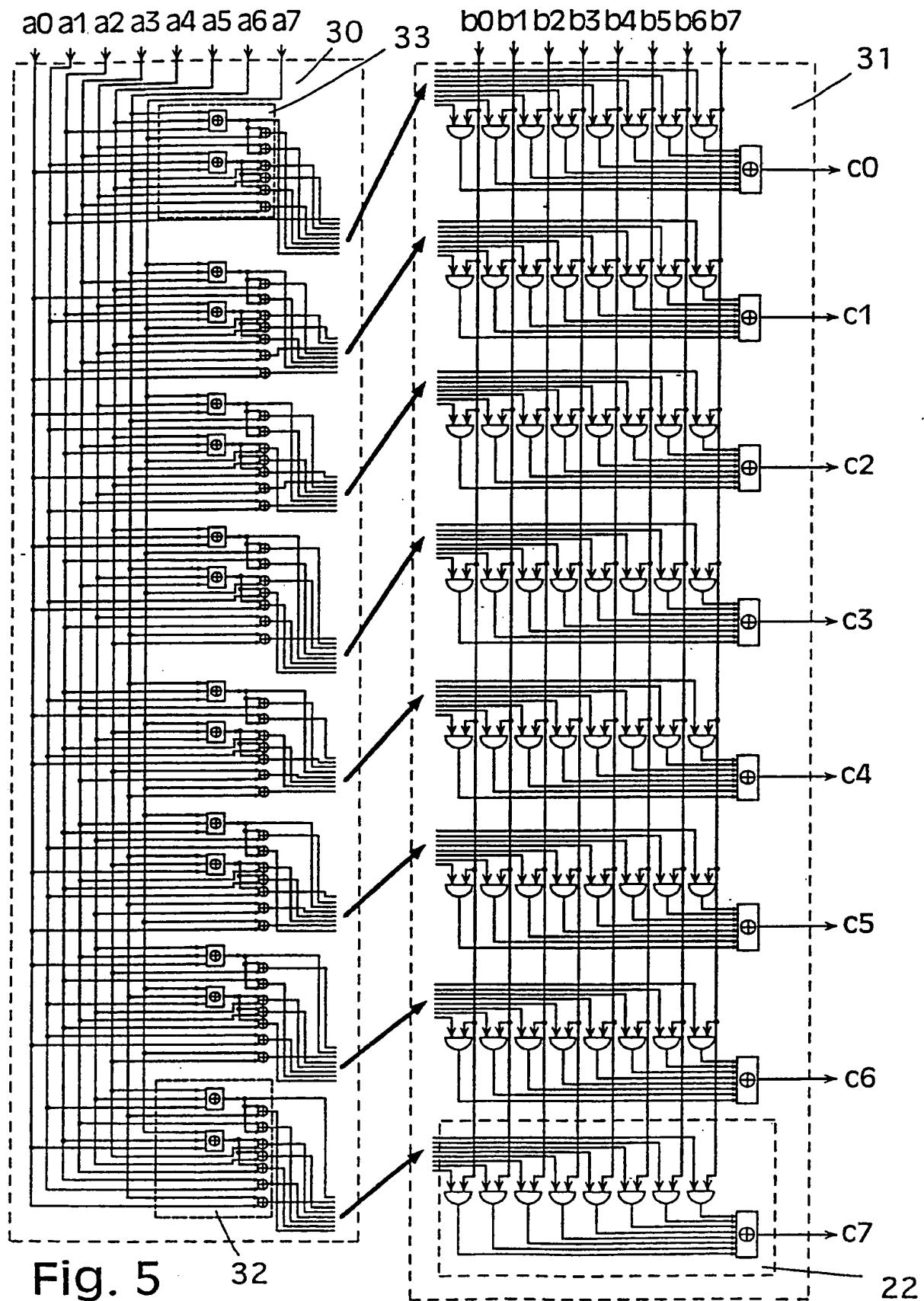


Fig. 4



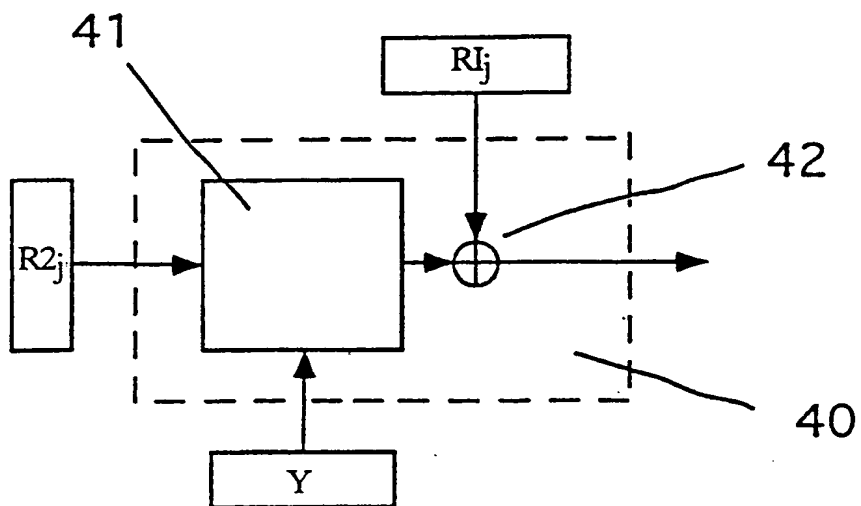


Fig. 6A

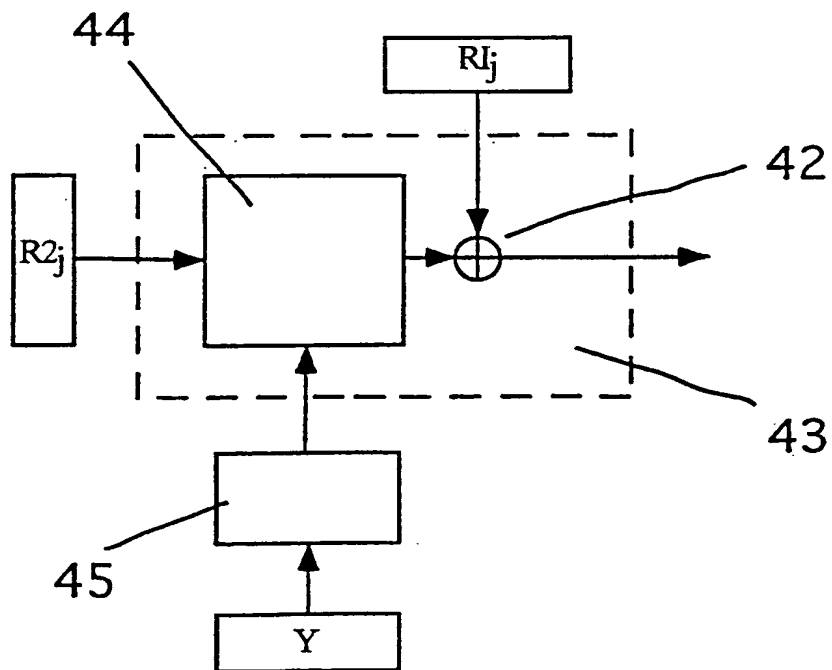


Fig. 6B

7/10

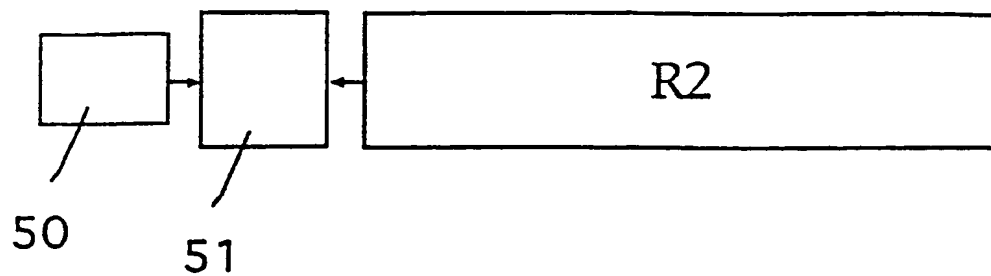


Fig. 7A

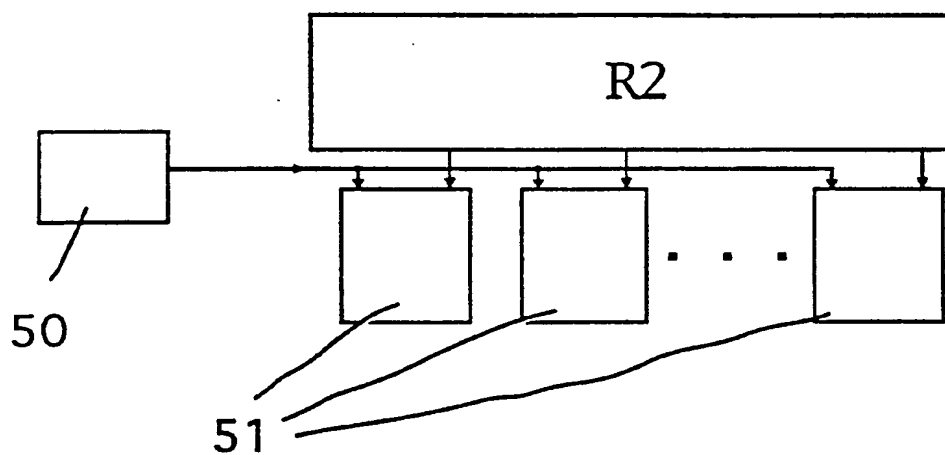


Fig. 7B

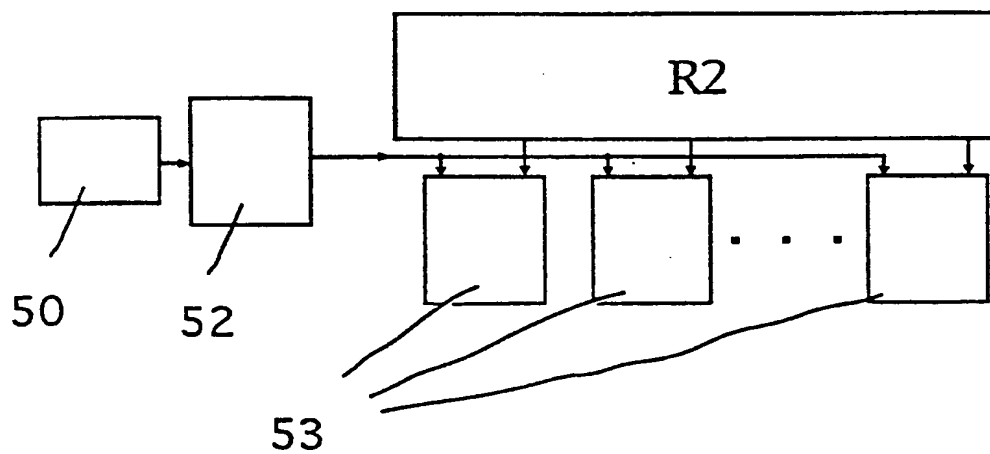


Fig. 7C

8/10

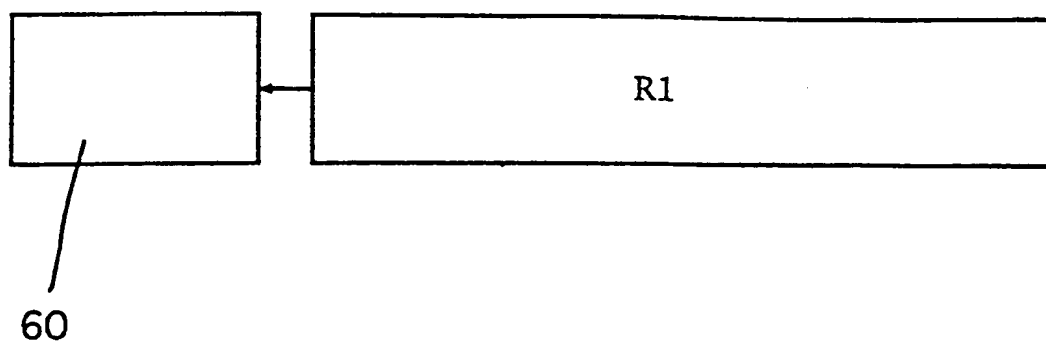


Fig. 8A

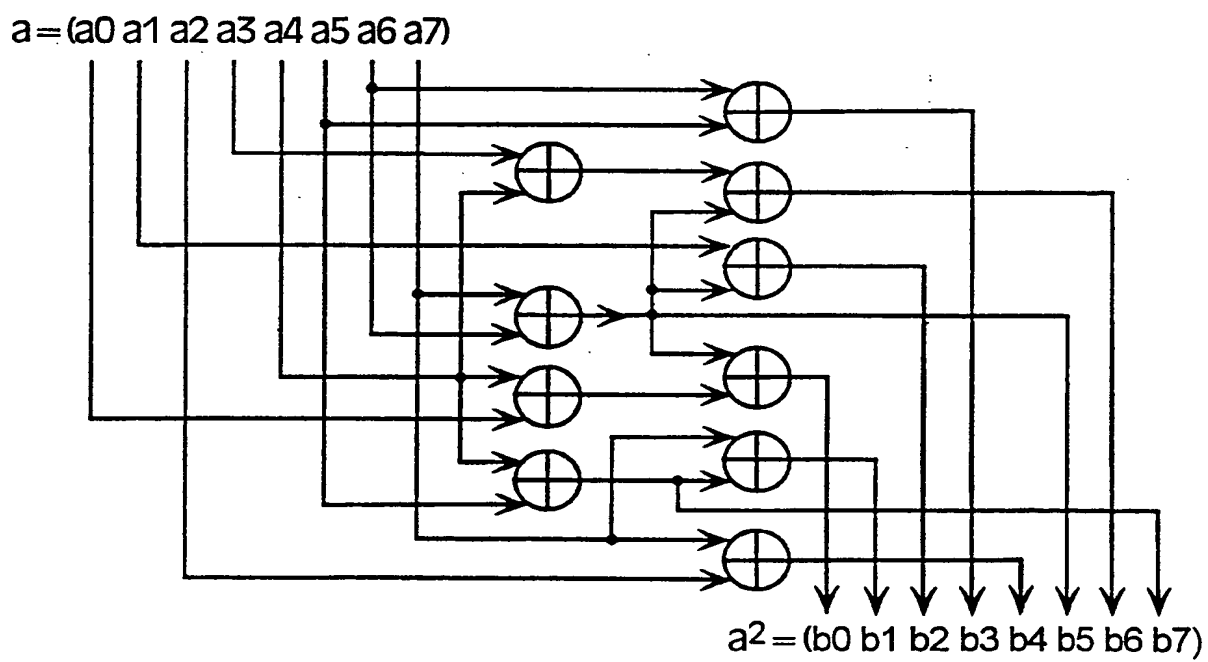


Fig. 8B

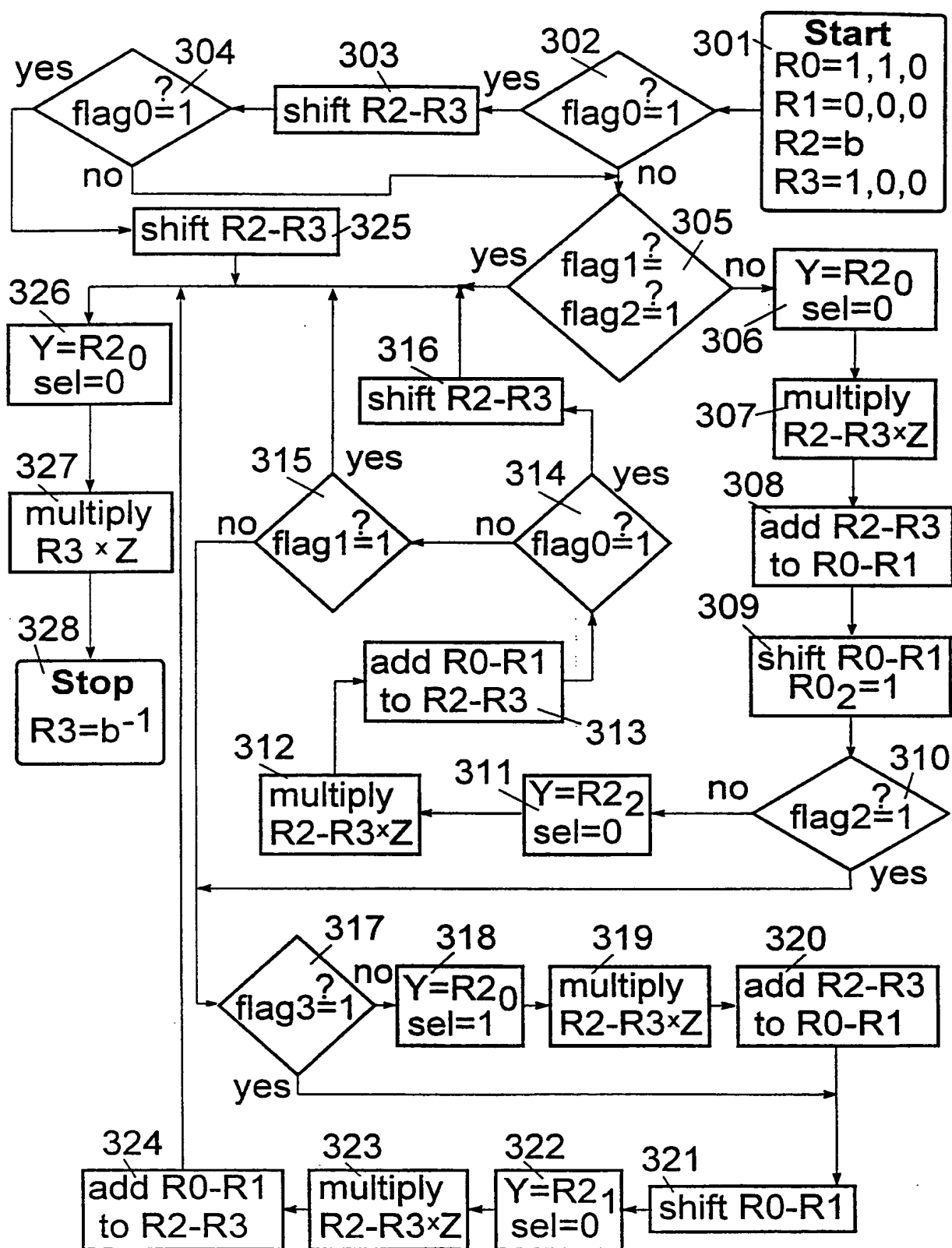


Fig. 9A

10/10

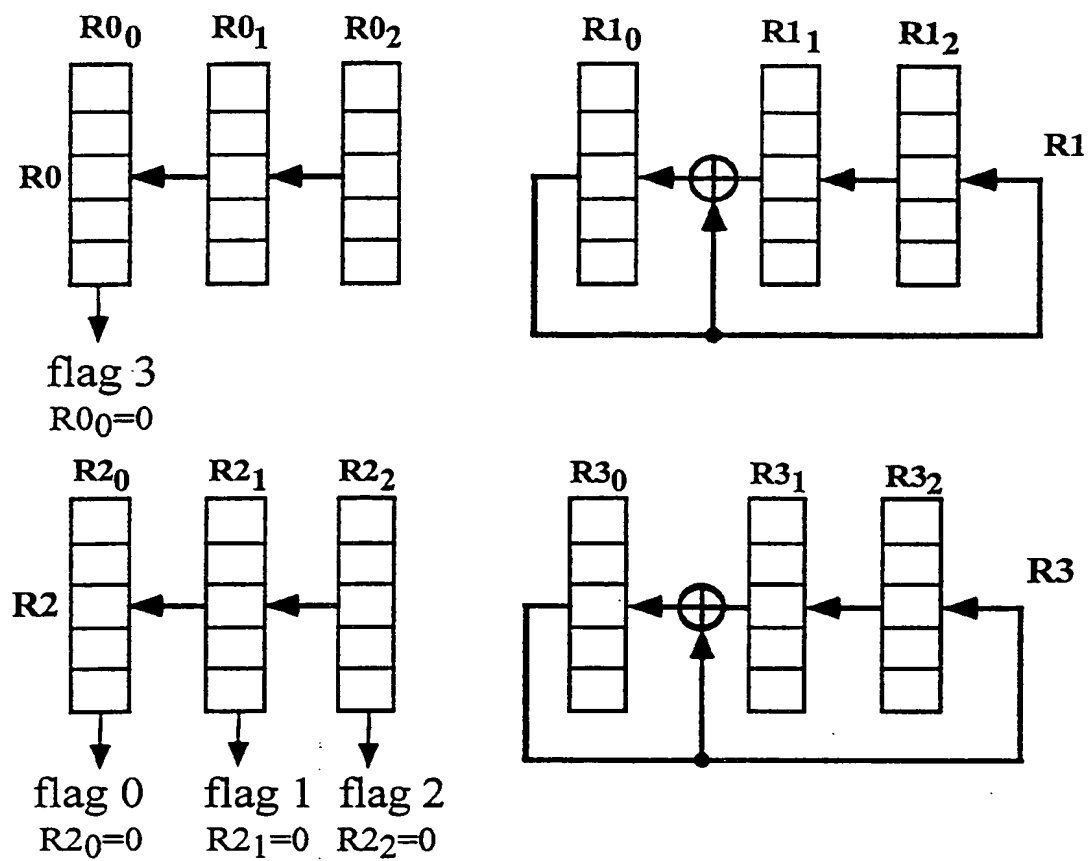


Fig. 9B

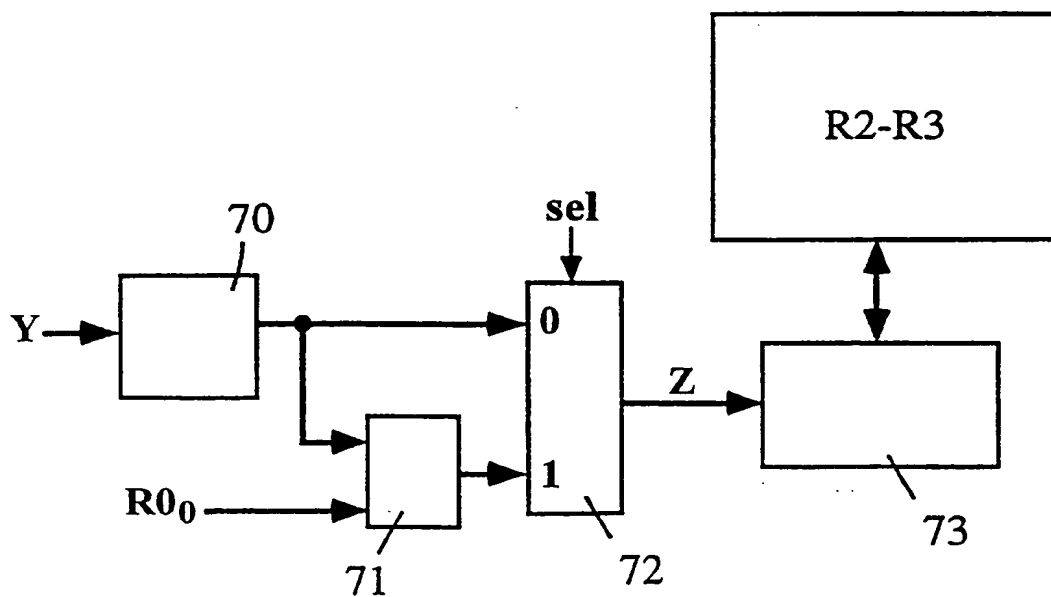


Fig. 9C

INTERNATIONAL SEARCH REPORT

International Application No

PCT/IL 98/00327

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F7/72

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	DE WIN E ET AL: "A fast software implementation for arithmetic operations in GF(2/sup n/)" ADVANCES IN CRYPTOLOGY - ASIACRYPT'96. PROCEEDINGS, KYONGJU, SOUTH KOREA, 3-7 NOV. 1996, pages 65-76, XP002081362 ISBN 3-540-61872-4, 1996, Berlin, Germany, Springer-Verlag, Germany cited in the application see page 68 - page 72 -----	1-11
A	EP 0 265 336 A (THOMSON CSF) 27 April 1988 see page 6, paragraph 2; figures 1,2,10 -----	1-11



Further documents are listed in the continuation of box C.



Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

20 October 1998

Date of mailing of the international search report

10/11/1998

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Verhoof, P

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No

PCT/IL 98/00327

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0265336 . A	27-04-1988	FR 2605769 A	29-04-1988
		DE 3778114 A	14-05-1992
		US 4852098 A	25-07-1989
<hr/>			

BEST AVAILABLE COPY